Uber Sign Up and Reporting System

Software Architecture Document

Version 1.0

Revision History

Date	Version	Description	Author
01/December/2020	1.0	Software Architecture Document	A. Arsiniega

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

1.2 Scope

This Software Architecture Document provides an architectural overview of an academic Uber Sign Up and Reporting System. The System is being developed by College of Charleston to study software architecture design and implementation as part of course CSIS 656.

1.3 Definitions, Acronyms and Abbreviations

See the Glossary [4].

2. Architectural Representation

This document presents the architecture as a series of views; use case view, logical view, process view and deployment view. There is no separate implementation view described in this document. These are views on an underlying Unified Modeling Language (UML) model.

Architectural Goals and Constraints

There are some key requirements and system constraints that have a significant bearing on the architecture. They are:

1. There are no database connections and many of these services like the RideService and the SignInService are no more than showing you the method or class you're in with no logic.
2. The SignUp and ReportWriterService however do make use of some logic by accepting and reporting on the different users of the system (Driver and Renter) and go further than applying discounts for users with a monthly subscription.
3. For SignUp we ensure that payment types are valid for all payments whether one-time or monthly before applying a 10% discount. This is achieved by validating the 12-digits for account numbers and 8-digits for routing numbers or checking valid gift cards that contain the 'UBER' string.

4. Use-Case View

1) As a driver (owner)
   I want to sign up using Uber Driver App
   So I can get an account

2) As a driver (owner)
   I want to sign in my account using Uber Driver App
   So I can be authenticated

3) As a driver (owner)
   I want to activate the app
   So that I can drive renters

4) As a driver (owner)
   I want to view a renter's location
   So I can decide whether to accept the request or not

5) As a driver (owner)
   I want to view a renter's trip estimate and earnings
   So I can decide whether to drive for the service

6) As a driver (owner)
   I want to accept to drive for the service
   So that I can earn money

7) As a driver (owner)
   I want to know the final earning estimate
   So that I know how much I earned

8) As a driver (owner)
   I want to be able to rate the renter
   So that the renter strives to be calm and courteous

9) As a renter
   I want to sign up using Uber App
   So I can get an account

10) As a renter
    I want to sign in using Uber App
    So I can login to my account

11) As a renter
    I want to see trip cost estimates
    So I can decide whether to rent the service

12) As a renter
    I want to accept to rent the service
    So that I can get to my destination

13) As a renter
    I want to be able to rate drivers
    So that the driver strives to provide quality service

14) As a renter
    I want to receive final trip estimate
    So that I know how much I owe

15) As a corporation
    I want to provide one uber driver app for driver sign ups
    So that I can confirm drivers

16) As a corporation
    I want to provide one uber app for renter sign ups
    So that I can confirm renters

17) As a corporation
    I want to boost prices during heavy traffic
    So that drivers can activate app and drive

18) As a corporation
    I want to review driving records periodically
    So that I ensure drivers are responsible

19) As a corporation
    I want to provide ratings for renters
    So that drivers can rate the renters

20) As a corporation
    I want to provide ratings for drivers
    So that renters can rate the service

21) As a corporation
    I want to authenticate drivers
    So that I can confirm drivers identity

22) As a corporation
    I want to add tips for my drivers
    So that renters who received great service can add an extra tip

23) As a renter
    I want to view different service types of vehicles
    So that I can have a better ride

24) As a renter
    I want to be able to add a tip
    So that I am provided a quality service

25) As a driver (owner)
    I want to be able to cancel on a renter
    So that I remain safe if there are any issues with the renter

26) As a renter
    I want to be able to cancel renting the service
    So that I remain safe if there are any issues with the driver

4.1 Architecturally-Significant Use Cases

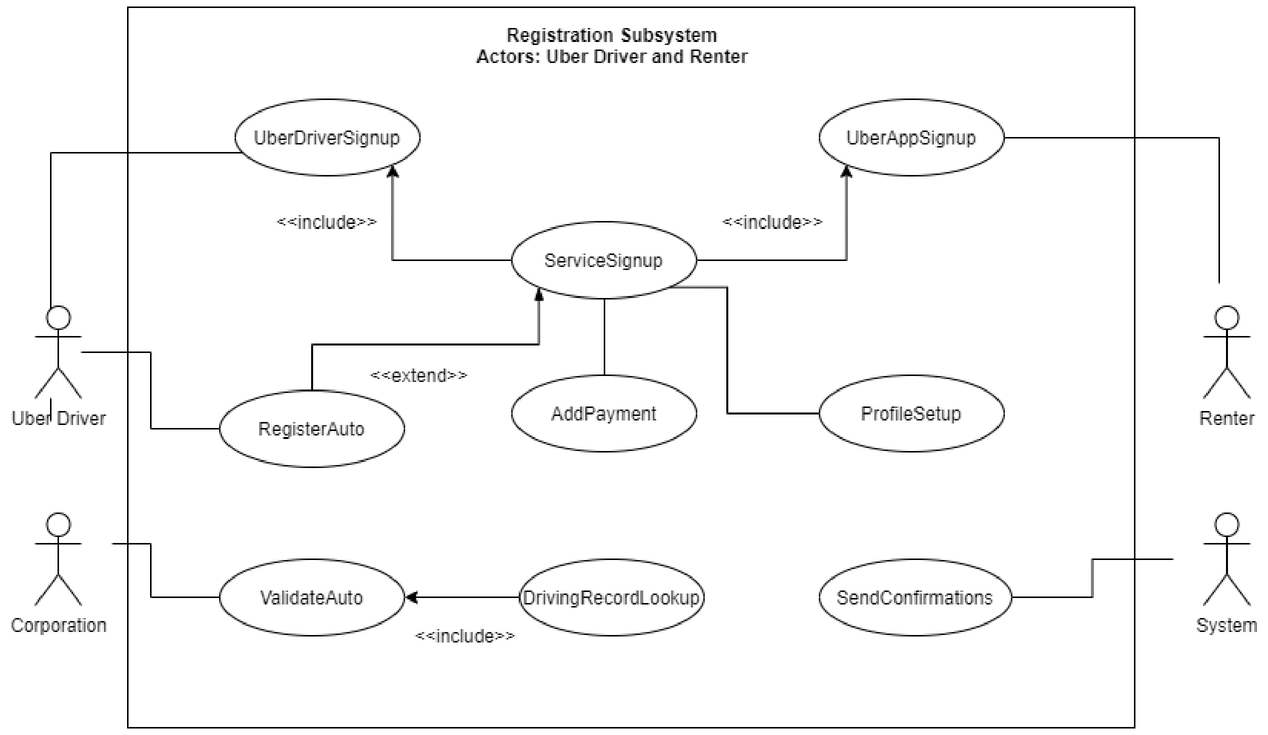

Diagram Name: Detailed Use Cases for Registration Subsystem

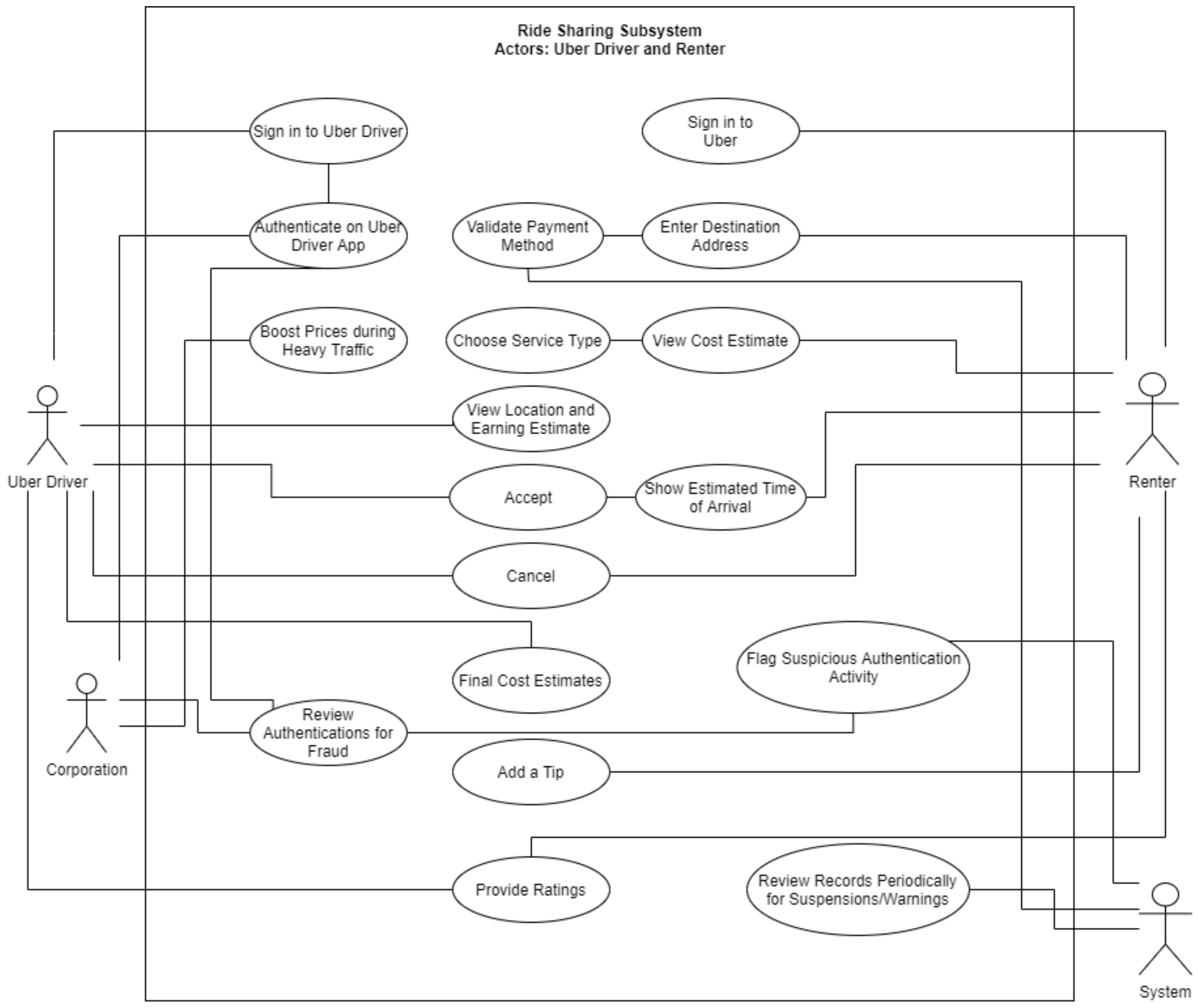
Ride Sharing Subsystem
Actors: Uber Driver and Renter
Sign in to Uber Driver
Sign in to Uber
Authenticate on Uber Driver App
Validate Payment Method
Enter Destination Address
Boost Prices during Heavy Traffic
Choose Service Type
View Cost Estimate
View Location and Earning Estimate
Uber Driver
Renter
Accept
Show Estimated Time of Arrival
Cancel
Final Cost Estimates
Flag Suspicious Authentication Activity
Corporation
Review Authentications for Fraud
Add a Tip
Provide Ratings
Review Records Periodically for Suspensions/Warnings
System

Diagram Name: New Use Cases for Ride Sharing Subsystem - All Actors

4.1.1 Uber Driver

Brief Description: This actor differs from the Renter by being the one providing the RideService. There is no payment required from the Driver like there is from the Renter. The Driver however can also be a user of the Uber service so they share many of the same details. The driver must be validated much more differently than just by the SignInService.

4.1.2 SignUpService

Brief Description: We make use of a User and Payment pairing to validate the first time sign up. A driver will have additional steps before they are able to actually sign in, like having their vehicle/insurance information validated by the system (periodically) and authenticating with the uber before the start of a work day.

4.1.3 RideService

Brief Description: A renter can initiate the DriveService to receive estimates of the cost and only once accepted will Drivers already authenticated in the system be notified.

4.1.4 ServiceFinderService

Brief Description: Because different types of Drivers (based on vehicle information) are available in different, usually larger markets, we must be able to display these to Renters so that they can choose the best option from Luxury vehicles to sharing the ride with 1-2 other people for a better rate..

4.1.5 ReportWriterService

Brief Description: The ReportWriterService will report on all users that have been through the SignUp service, all their payment information, users with valid monthly discounts, users with invalid monthly discounts, users with valid one-time information, and finally those with invalid one-time payment information.

5. Logical View

The logical view of the my-uber application is comprised of the 3 main packages: Services, Models, and the Uber package which contains App.java.

The logical view of the uber-service application is mostly comprised of the main Uber package and the UberService.java. However, we’ve also successfully integrated all the packages from my-uber here.

The Services Package contains control classes for interfacing with the payment system, sign up system, and reporting system.

The Models Package contains classes for each of the forms that the actors use to communicate with the System. Boundary classes exist to support payment and sign up.

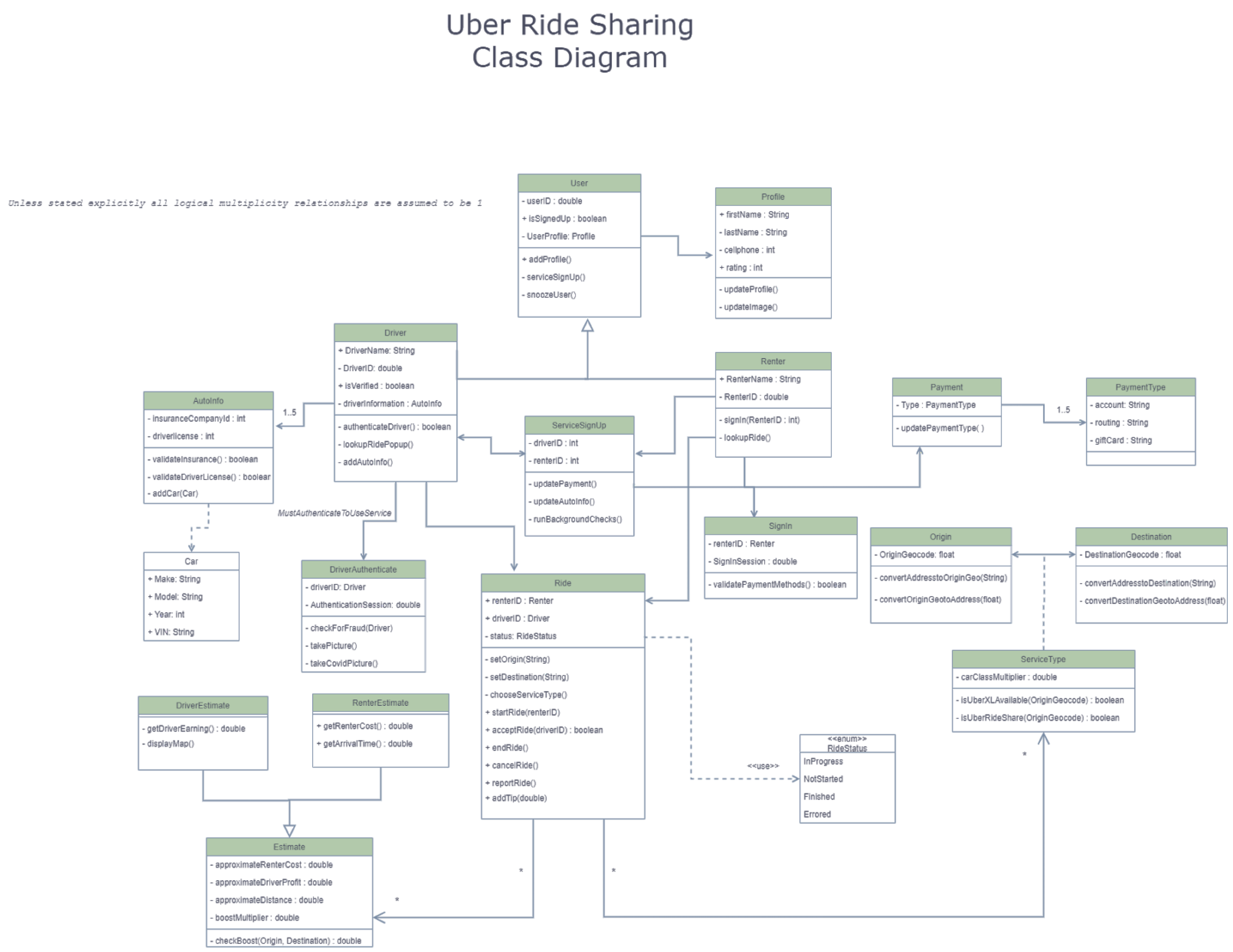

Diagram Name: Uber Ride Sharing Class Diagram

6. Process View

The main processing is done in the Uber package by making use of the SignUpService. There is no threading or controller. The uber-service makes use of a RESTful architecture to display the users of the Uber system. The ability to add a user with a REST call or an individual user by id is also available: